Session 3.2

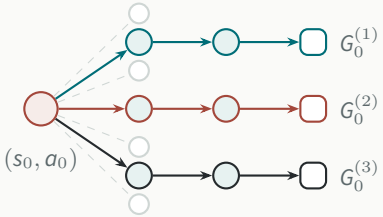
Monte Carlo: Evaluation, Control, and the Data



Zhigang Feng

© Zhigang Feng
Redistribution prohibited without permission

Basic Monte Carlo: Walk N Paths from One Cell, Average the Returns



Basic Monte Carlo for one cell

input: g, (s, a), N paths, horizon T
for $i = 1, \ldots, N$:
  simulate T periods: $a_0 = a$, then g
  $G_0^{(i)} \leftarrow \sum_{j=0}^{T-1} \gamma^j r_{j+1}$
return $\widehat{Q}^g(s, a) \leftarrow \frac{1}{N} \sum_i G_0^{(i)}$

- **Monte Carlo** (MC): without P_z, simulate N lifetimes from (s, a), Deck 3.1's walk run to the end with no $\widehat{V}$ to end on, and average their discounted utility.
- Each path is cut at $T = 300$ periods, where the discounted tail no longer matters; the test bed is the smallest instance, the 7×2 grid.
- **Cost per cell $O(NT)$, whatever the number of states or shock histories.**

© Zhigang Feng · Redistribution prohibited without permission

The Exact Q^* Table: the Answer Key

$k \setminus k'$	1.50	2.25	3.00	3.75	4.50	5.25	6.00
1.50	**0.563**	-0.004	–	–	–	–	–
2.25	1.226	**1.932**	1.441	–	–	–	–
3.00	1.609	2.533	**2.946**	2.414	–	–	–
3.75	1.879	2.898	3.511	**3.735**	3.054	–	–
4.50	2.088	3.160	3.864	**4.277**	4.275	3.620	–
5.25	2.259	3.366	4.122	4.623	**4.803**	4.784	4.126
6.00	2.403	3.534	4.325	4.877	5.144	**5.303**	5.257

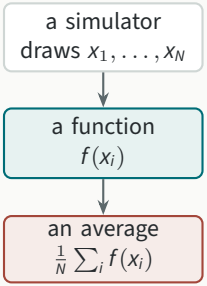
Redistribution prohibited without permission

- The $z = 0.9$ half of the 7×2 grid, from DP. Bold: the row maximum, the optimal k': Deck 3.1's answer key.
- Two rows to look at. At $k = 3.75$ the best choice wins by 0.224, 1.13% of consumption for ever; at $k = 4.50$ by 0.002, a hair. **Big gaps are easy to learn; hair-thin gaps are what sampling gets wrong, and they cost little.**

© Zhigang Feng

Source: Session 3 lab, block A: the 7×2 grid, $\gamma = 0.95$; in consumption, $\exp\{(1-\gamma)\Delta\} - 1$.

What Monte Carlo Is: Simulate and Average



shocks, paths, households

a moment, a return

$\rightarrow E[f(X)]$

- One simulated life is luck; average many and it cancels: the mean tends to $E[f(X)]$.
- Its error falls as $1/\sqrt{N}$, whatever the number of states; close values need many paths.
- **Economists already use it**: the simulated method of moments, same estimator, another f.
- Here f is a lifetime return, and the average improves a *rule*, not a parameter.
- **The estimator is old. New is what gets averaged, and what for.**

© Zhigang Feng Redistribution prohibited without permission

Dynamic Programming or Monte Carlo: When to Use Which

	Dynamic programming	**Monte Carlo**
needs	P, summed at every state	a simulator: only paths
updates	every state, every sweep	visited cells, after a path ends
error	none from sampling	variance, falling like $1/\sqrt{N}$
cost, d variables	m^d states, a sum at each	N paths a cell at any d; still a cell per pair

- **Sampling replaces the sum over tomorrow, not the table**: one cell costs the same at any d, but there is still a cell per state–action pair (Deck 3.4 drops the table).
- **Stay with DP** when the state can be swept and the expectation summed; **leave it when either fails**.
- From here to Deck 3.4's model-free learners we pretend not to know P_z; in Session 4 the distribution's law of motion is usually truly unknown to the household.
- **Model-free is not error-free: a misspecified simulator passes its error straight on.**

© Jiaqing Feng · Redistribution prohibited without permission

Session 3.2

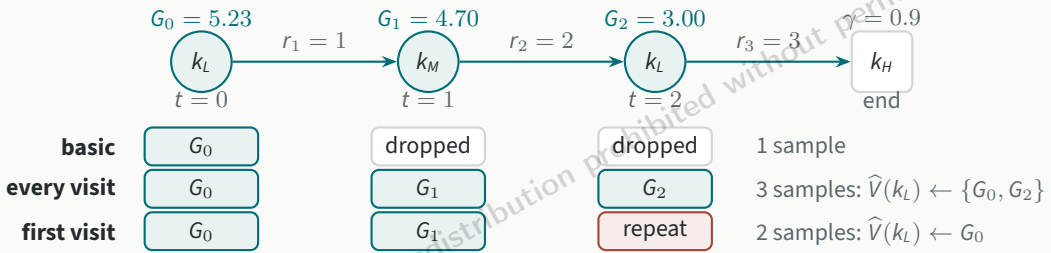
One Path, Many Samples

Reuse every visit a path makes, and pay for it in coverage



© Zhigang Feng Redistribution prohibited without permission

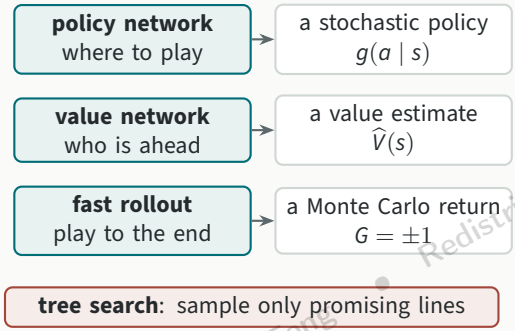
One Path Updates Every Cell It Visits, and the Price



© Zhigang Feng Redistribution prohibited without permission

- The rest of a path samples every state it passes through: one path updates many cells.
- **First-visit**: a state counts once per path; its estimate is the mean of its $C(s)$ returns, the **visit count**. This session uses it.
- The price: a cell gets data only where the rule walks. Harmless for valuing a fixed rule, decisive for improving one: nothing is learned about actions it never takes.
- **A second price: a whole return is noisy and arrives late. Deck 3.3 changes both.**

AlphaGo: a Policy, a Value, and Monte Carlo Returns



- Go: about 2.1×10^{170} legal positions, some 250 moves per turn. Ten moves ahead is $250^{10} \approx 10^{24}$ lines.
- The rules are known. A full backup is still impossible.
- Policy network: trained on 30 million human positions, then improved by self-play.
- **A policy, a value and returns are this session's objects; tree search is not. None enumerates the positions.**

© Zhigang Feng · Redistribution prohibited without permission

Source: Positions: J. Tromp, 2016; AlphaGo: D. Silver et al., 2016.

Session 3.2

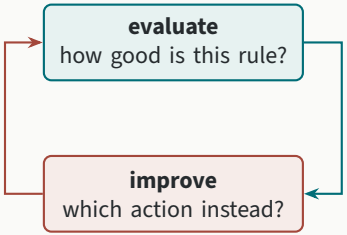
From Evaluation to Control

Improve the rule from its estimated values, and find out what estimation breaks



© Zhigang Feng Redistribution prohibited without permission

Evaluation Was the Easy Half. Now Improve

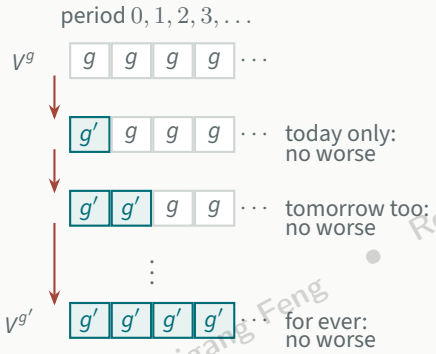


this deck so far:
the top box only

- Everything so far values a *fixed* rule. Finding a better rule from the estimates is **control**.
- The recipe is policy iteration, value the rule then improve it, now on estimated values.
- Any alternation of the two, each possibly incomplete, is **generalised policy iteration**.
- **Next: improvement is safe on exact values; estimates break it (p. 12–14).**

© Zhigang Feng
Redistribution prohibited without permission

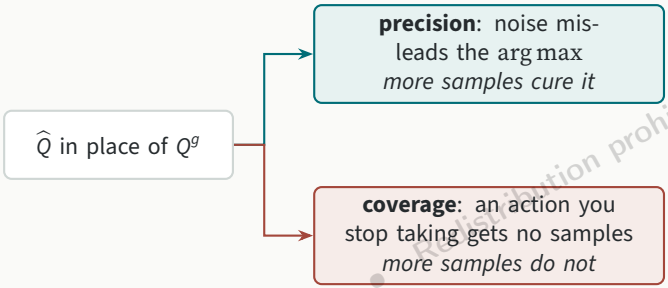
Improving the Rule Cannot Hurt: the One-Shot Deviation Principle



- **Deviate for one period**: take $g'(s)$ today, then follow g. If that is no worse at every state, $Q^g(s, g'(s)) \geq V^g(s)$, the switch cannot hurt.
- **Tomorrow is another today**, wherever g' has led: deviate again. Each row is no worse; with discounting, $V^{g'} \geq V^g$.
- The **one-shot deviation principle**: if no one-period deviation improves a rule, it is optimal.
- **It needs the exact Q^g; Monte Carlo hands us an estimate.**

© Zhigang Feng
Redistribution of courses without permission

Estimation Breaks the Improvement Theorem — Two Ways



$\widehat{Q}$ in place of Q^g

precision: noise misleads the $\arg\max$ *more samples cure it*

a finite-sample problem

coverage: an action you stop taking gets no samples *more samples do not*

a structural problem

© Zhigang Tang • Redistribution prohibited without permission

- The theorem promised: greedy on the *exact* Q^g is never worse. With $\widehat{Q}$ it fails in two ways, needing different cures.
- **One creates mistakes; the other can make them permanent.** A learner that sees only where its own rule goes meets both. The next two pages put numbers on each.

(1) Precision: When Noise Wins the $\arg\max$

- Each cell's estimate is a sample mean with an error of size $\sigma_G/\sqrt{C(s,a)}$: $\widehat{Q}(s,a) = Q^g(s,a) + e_a$. The luckiest estimate wins the $\arg\max$: the **winner's curse** (Deck 3.3's maximisation bias).
- Best action a^*, a rival a, gap $\Delta = Q^g(s,a^*) - Q^g(s,a)$: greedy errs when $e_a - e_{a^*} > \Delta$. With C returns a cell and Φ the normal cdf,

$$P(\text{wrong pick}) \approx \Phi\big(-\Delta\sqrt{C}\,/\,\sigma_G\sqrt{2}\,\big).$$

- One ratio decides, the gap in units of the noise. Ranking right 19 times in 20 takes $C \approx 5.4\,\sigma_G^2/\Delta^2$ returns: halve the gap, four times the data.
- Near the best k' the value is flat, $Q^g(s,a) \approx Q^g(s,a^*) - \frac{1}{2}|Q_{aa}|(a-a^*)^2$, so gaps are small: neighbours are the hardest to rank and the cheapest to confuse. The expected loss $\Delta \cdot P(\text{wrong})$ is at most $0.24\,\sigma_G/\sqrt{C}$, whatever the gap.
- **Small gaps and few samples: the $\arg\max$ picks noise. The error, $\Phi(-\Delta\sqrt{C}/\sigma_G\sqrt{2})$, fades as the count grows.**

© Zhigang Feng • Redistribution prohibited without permission

(2) Coverage: the Actions You Stop Taking

- A cell moves only when visited, $\widehat{Q}(s,a) \leftarrow \widehat{Q}(s,a) + \big(G - \widehat{Q}(s,a)\big)/C(s,a)$; its error $\sigma_G/\sqrt{C(s,a)}$ falls only if $C(s,a) \to \infty$.
- Greedy takes a only while $\widehat{Q}(s,a)$ is the row's maximum. Let the best action a^* draw an unlucky first return, $G_1 < \widehat{Q}(s,a')$ for a rival a', which has positive probability: $C(s,a^*)$ stays at 1 and, whatever the number of paths,

$$Q^g(s,a^*) - \widehat{Q}(s,a^*) = Q^g(s,a^*) - G_1 \quad \text{for ever.}$$

- Identification needs **overlap**: $E[G \mid s,a]$ is estimable only if the data's rule takes a at s with positive probability; greedy sets $P(a \mid s) = 0$ for all but one action.
- Rothschild's firm (a **multi-armed bandit**: one state, several prices, unknown demand) never learns a price it stops charging, and can settle on the wrong one for ever.
- **The cure: a rule with $g(a \mid s) \geq \epsilon/|\mathcal{A}(s)|$ visits every cell again and again, so $\widehat{Q} \to Q^g$. Precision errors fade as $\sigma_G/\sqrt{C}$; coverage errors stay, unless every action keeps a positive probability.**

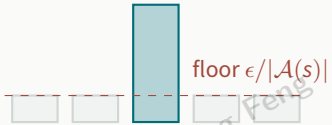
© Zhigang Feng • Redistribution prohibited without permission

Two Cures for Coverage: a Reset Button, or a Trembling Hand

exploring starts: a reset button



ϵ-**greedy**: a trembling hand



every k' keeps a floor; the best-looking gets the rest

- **Exploring starts**: a random (s_0, a_0), then g. A reset button only a simulator has.
- ϵ-**greedy**: mostly the best-looking action, with probability ϵ any one. No reset.
- A state visited again and again has every action tried again and again: coverage restored.
- The action is k': a tremble moves the economy to states the rule alone would not visit.
- **Two prices: you learn the value of a rule that trembles, and only where it goes.**

© Zhiyang Feng · Redistribution prohibited without permission

ϵ-Greedy Monte Carlo Control

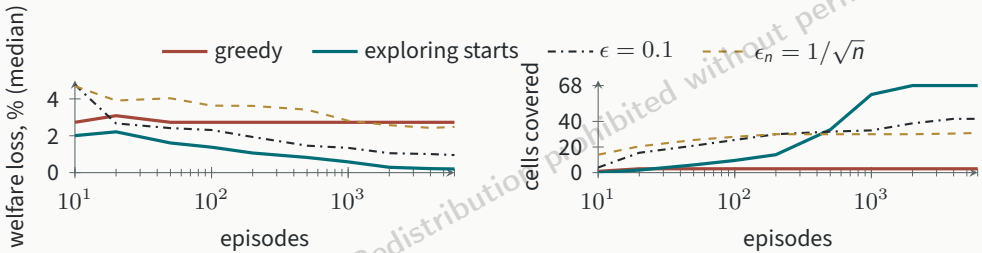
First-visit Monte Carlo control, on the rule's own data

1 initialise $\widehat{Q}$, counts $C \leftarrow 0$; choose a start s_0 (it may be fixed) and a schedule ϵ_i
2 **for** episode $i = 1, \ldots, N$:
3 from s_0, draw every action from g_{ϵ_i} built on the current $\widehat{Q}$; simulate T periods
4 $G \leftarrow 0$; **for** $t = T-1, \ldots, 0$: $G \leftarrow r_{t+1} + \gamma G$
5 first visit of $x = (s_t, a_t)$, first half only: $C(x) \leftarrow C(x) + 1$, $\widehat{Q}(x) \leftarrow \widehat{Q}(x) + \big(G - \widehat{Q}(x)\big)/C(x)$
6 **return** $\widehat{Q}$ and the greedy rule $g(s) = \arg\max_a \widehat{Q}(s, a)$ for deployment

- With exploring starts instead, line 3 draws (s_0, a_0) at random and then follows the greedy rule.
- The rule that collects the data trembles. The rule deployed at the end does not.
- **Trembling, the rule improves to the best trembling rule; to reach g^* the trembles must fade (Deck 3.3). Line 5 is a running mean: Deck 3.3 starts from that line.**

© Zhigang Feng Redistribution prohibited without permission

Measured on the Whole Model: Greedy, Exploring Starts, ϵ-Greedy



Redistribution prohibited without permission

- After 6,000 episodes greedy covers three cells (ten samples or more) and loses most; exploring starts covers all 68 and loses least; $\epsilon = 0.1$ covers 42, in between; $\epsilon_n = 1/\sqrt{n}$ has the right limit but explores too little (0.1 by episode 100).
- Loss: the welfare loss in % of consumption, Deck 3.1's grade, a plain mean over the 14 states.

Source: Recorded runs of the book's own scripts: medians of 20 seeds, one path per cell to start; 7×2 model, $\gamma = 0.95$.

Session 3.2

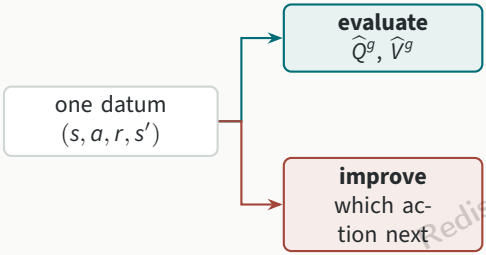
Who Generates the Data

What a datum is, what it is for, and whose value you end up with



© Zhigang Feng · Redistribution prohibited without permission

What the Data Are, and What They Are For



- A datum is one tuple doing two jobs; every method this session chooses how to do each.
- On-policy or off-policy is one question about that pair: **is the rule that generated the tuple the rule being valued?**
- The economist's version: a new Taylor rule, valued on the Fed's post-1979 data.
- **Ask of any result: which rule produced these data, and whose value is being reported?**

© Zhigang Feng
Redistribution prohibited without permission

Importance Sampling: Reweighting Someone Else's Data

- **The question.** The paths were walked by the trembling rule b. Can they value the greedy rule g instead?
- Yes, by inverse probability weighting: weight each return by how much likelier g makes its path, the **importance ratio** $\rho = \prod_j g(a_j \mid s_j)/b(a_j \mid s_j)$.
- The shocks have the same probabilities under both rules and cancel: ρ needs no P_z.
- A greedy g takes one action per state, so one tremble zeroes a path's weight and none inflates it: on long paths most weights are zero, a few huge.
- It needs overlap: b must keep trying every action g might take.
- **The ratio is a product over the whole path, and that is the problem: Q-learning (Deck 3.3) is off-policy one step at a time, with no product.**

© Zhigang Feng — Redistribution prohibited without permission

On-Policy and Off-Policy, in the Algorithm

Monte Carlo from ϵ-greedy data, two ways

1 initialise $\widehat{Q}$; the **behaviour rule** $b = g_\epsilon$, built on $\widehat{Q}$, collects the data
2 **for** each episode: simulate T periods, every action drawn from b
3 $G \leftarrow 0$; **for** $t = T-1, \ldots, 0$: $G \leftarrow r_{t+1} + \gamma G$; at first visits only:
4 **on-policy:** $\widehat{Q}(s_t, a_t) \leftarrow$ running mean of G (the value of b itself)
5 **off-policy:** $\rho \leftarrow \prod_{j>t} g(a_j \mid s_j)/b(a_j \mid s_j)$, with g greedy in $\widehat{Q}$
6 $\widehat{Q}(s_t, a_t) \leftarrow$ mean of G weighted by ρ (the value of g)
7 **return** the greedy rule in $\widehat{Q}$

- The same paths, two **target rules**: line 4 values the rule that collected the data, trembles included (**on-policy**); lines 5–6 reweight by the previous page's ρ and value the greedy rule g (**off-policy**).
- This deck's two controls were on-policy. **Lines 4 and 5–6 are the whole difference.**

© Shuguang Zhang
Redistribution prohibited without permission

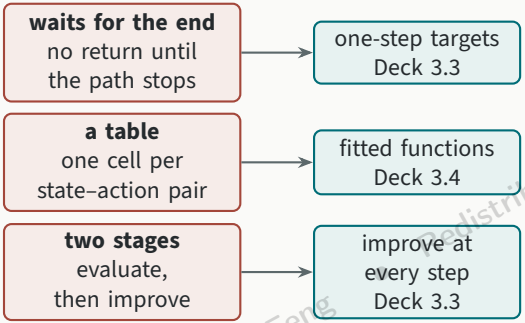
On or Off: the Trade-off

	on-policy: ϵ-greedy MC control	**off-policy**: MC with importance sampling
whose value	the trembling rule you ran	the greedy rule, from the trembling data
each path	counts as it is	reweighted: one tremble zeroes it, none inflates it
noise	an average's	a product's, growing with the path
to reach g^*	trembles must fade, but slowly	any rule still trying every action of g
old data	no: the current rule's only	yes, with known probabilities covering g

- **On-policy: an accurate value of a slightly wrong rule, the trembling one. Off-policy: a noisy value of the right rule, from any rule's data.**
- A new Taylor rule on old-regime data is off-policy; huge weights mean thin overlap.
- Reweighting assumes only actions differ; Lucas: the economy's responses change too.
- Trembles are free in a simulator, costly in an economy: real data push off-policy.
- Whole returns make off-policy dear; one-step targets make it cheap (Deck 3.3).

© Zhiqiang Feng. Redistribution prohibited without permission

What Tabular Monte Carlo Cannot Do



© Zhigang Feng
Redistribution prohibited without permission

- Three limits, each naming what comes next: waiting for the path's end; a table in high dimension; evaluation and improvement in two stages.
- The third is a way of teaching the method, not a requirement: SARSA and Q-learning improve at every step.
- Beyond the three limits, one cost: the variance of whole-path returns (p. 7); one-step targets cut it too.
- **Next, Deck 3.3: update after one step, and stop waiting for the path to end.**